Our choice of the parameters $b$ and $r$, combined with (13.44), now implies the following.

**Proof.** We have just argued that the probability of the bad event $\mathcal{E}$ is very small, at most $(mN)^{-(z-1)}$ for an arbitrarily large constant $z$. And provided that $\mathcal{E}$ does not happen, (13.47) tells us that the duration of the schedule is bounded by

$$b(r+d) = \frac{3c}{r}(r+d) = 3c + d \cdot \frac{3c}{r} = 3c + d(3q \log(mN)) = O(c + d \log(mN)).$$

■

## 13.12 Background: Some Basic Probability Definitions

For many, though certainly not all, applications of randomized algorithms, it is enough to work with probabilities defined over finite sets only; and this turns out to be much easier to think about than probabilities over arbitrary sets. So we begin by considering just this special case. We'll then end the section by revisiting all these notions in greater generality.

### Finite Probability Spaces

We have an intuitive understanding of sentences like, "If a fair coin is flipped, the probability of 'heads' is 1/2." Or, "If a fair die is rolled, the probability of a '6' is 1/6." What we want to do first is to describe a mathematical framework in which we can discuss such statements precisely. The framework will work well for carefully circumscribed systems such as coin flips and rolls of dice; at the same time, we will avoid the lengthy and substantial philosophical issues raised in trying to model statements like, "The probability of rain tomorrow is 20 percent." Fortunately, most algorithmic settings are as carefully circumscribed as those of coins and dice, if perhaps somewhat larger and more complex.

To be able to compute probabilities, we introduce the notion of a *finite probability space*. (Recall that we're dealing with just the case of finite sets for now.) A finite probability space is defined by an underlying *sample space* $\Omega$, which consists of the possible *outcomes* of the process under consideration. Each point $i$ in the sample space also has a nonnegative *probability mass* $p(i) \geq 0$; these probability masses need only satisfy the constraint that their total sum is 1; that is, $\sum_{i \in \Omega} p(i) = 1$. We define an *event* $\mathcal{E}$ to be any subset of

$\Omega$—an event is defined simply by the set of outcomes that constitute it—and we define the *probability* of the event to be the sum of the probability masses of all the points in $\mathcal{E}$. That is,

$$\Pr[\mathcal{E}] = \sum_{i \in \mathcal{E}} p(i).$$

In many situations that we'll consider, all points in the sample space have the same probability mass, and then the probability of an event $\mathcal{E}$ is simply its size relative to the size of $\Omega$; that is, in this special case, $\Pr[\mathcal{E}] = |\mathcal{E}|/|\Omega|$. We use $\overline{\mathcal{E}}$ to denote the complementary event $\Omega - \mathcal{E}$; note that $\Pr\left[\overline{\mathcal{E}}\right] = 1 - \Pr[\mathcal{E}]$.

Thus the points in the sample space and their respective probability masses form a complete description of the system under consideration; it is the events—the subsets of the sample space—whose probabilities we are interested in computing. So to represent a single flip of a "fair" coin, we can define the sample space to be $\Omega = \{\texttt{heads}, \texttt{tails}\}$ and set $p(\texttt{heads}) = p(\texttt{tails}) = 1/2$. If we want to consider a biased coin in which "heads" is twice as likely as "tails," we can define the probability masses to be $p(\texttt{heads}) = 2/3$ and $p(\texttt{tails}) = 1/3$. A key thing to notice even in this simple example is that defining the probability masses is a part of defining the underlying problem; in setting up the problem, we are specifying whether the coin is fair or biased, not deriving this from some more basic data.

Here's a slightly more complex example, which we could call the *Process Naming*, or *Identifier Selection Problem*. Suppose we have $n$ processes in a distributed system, denoted $p_1, p_2, \ldots, p_n$, and each of them chooses an identifier for itself uniformly at random from the space of all $k$-bit strings. Moreover, each process's choice happens concurrently with those of all the other processes, and so the outcomes of these choices are unaffected by one another. If we view each identifier as being chosen from the set $\{0, 1, 2, \ldots, 2^k - 1\}$ (by considering the numerical value of the identifier as a number in binary notation), then the sample space $\Omega$ could be represented by the set of all $n$-tuples of integers, with each integer between 0 and $2^k - 1$. The sample space would thus have $(2^k)^n = 2^{kn}$ points, each with probability mass $2^{-kn}$.

Now suppose we are interested in the probability that processes $p_1$ and $p_2$ each choose the same name. This is an event $\mathcal{E}$, represented by the subset consisting of all $n$-tuples from $\Omega$ whose first two coordinates are the same. There are $2^{k(n-1)}$ such $n$-tuples: we can choose any value for coordinates 3 through $n$, then any value for coordinate 2, and then we have no freedom of choice in coordinate 1. Thus we have

$$\Pr[\mathcal{E}] = \sum_{i \in \mathcal{E}} p(i) = 2^{k(n-1)} \cdot 2^{-kn} = 2^{-k}.$$

This, of course, corresponds to the intuitive way one might work out the probability, which is to say that we can choose any identifier we want for process $p_2$, after which there is only 1 choice out of $2^k$ for process $p_1$ that will cause the names to agree. It's worth checking that this intuition is really just a compact description of the calculation above.

## Conditional Probability and Independence

If we view the probability of an event $\mathcal{E}$, roughly, as the likelihood that $\mathcal{E}$ is going to occur, then we may also want to ask about its probability given additional information. Thus, given another event $\mathcal{F}$ of positive probability, we define the *conditional probability of* $\mathcal{E}$ *given* $\mathcal{F}$ as

$$\Pr[\mathcal{E} \mid \mathcal{F}] = \frac{\Pr[\mathcal{E} \cap \mathcal{F}]}{\Pr[\mathcal{F}]}.$$

This is the "right" definition intuitively, since it's performing the following calculation: Of the portion of the sample space that consists of $\mathcal{F}$ (the event we "know" to have occurred), what fraction is occupied by $\mathcal{E}$?

One often uses conditional probabilities to analyze $\Pr[\mathcal{E}]$ for some complicated event $\mathcal{E}$, as follows. Suppose that the events $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_k$ each have positive probability, and they partition the sample space; in other words, each outcome in the sample space belongs to exactly one of them, so $\sum_{j=1}^{k} \Pr[\mathcal{F}_j] = 1$. Now suppose we know these values $\Pr[\mathcal{F}_j]$, and we are also able to determine $\Pr[\mathcal{E} \mid \mathcal{F}_j]$ for each $j = 1, 2, \ldots, k$. That is, we know what the probability of $\mathcal{E}$ is if we assume that any one of the events $\mathcal{F}_j$ has occurred. Then we can compute $\Pr[\mathcal{E}]$ by the following simple formula:

$$\Pr[\mathcal{E}] = \sum_{j=1}^{k} \Pr[\mathcal{E} \mid \mathcal{F}_j] \cdot \Pr[\mathcal{F}_j].$$

To justify this formula, we can unwind the right-hand side as follows:

$$\sum_{j=1}^{k} \Pr[\mathcal{E} \mid \mathcal{F}_j] \cdot \Pr[\mathcal{F}_j] = \sum_{j=1}^{k} \frac{\Pr[\mathcal{E} \cap \mathcal{F}_j]}{\Pr[\mathcal{F}_j]} \cdot \Pr[\mathcal{F}_j] = \sum_{j=1}^{k} \Pr[\mathcal{E} \cap \mathcal{F}_j] = \Pr[\mathcal{E}].$$

***Independent Events*** Intuitively, we say that two events are *independent* if information about the outcome of one does not affect our estimate of the likelihood of the other. One way to make this concrete would be to declare events $\mathcal{E}$ and $\mathcal{F}$ independent if $\Pr[\mathcal{E} \mid \mathcal{F}] = \Pr[\mathcal{E}]$, and $\Pr[\mathcal{F} \mid \mathcal{E}] = \Pr[\mathcal{F}]$. (We'll assume here that both have positive probability; otherwise the notion of independence is not very interesting in any case.) Actually, if one of these two equalities holds, then the other must hold, for the following reason: If $\Pr[\mathcal{E} \mid \mathcal{F}] = \Pr[\mathcal{E}]$, then

$$\frac{\Pr[\mathcal{E} \cap \mathcal{F}]}{\Pr[\mathcal{F}]} = \Pr[\mathcal{E}],$$

and hence $\Pr[\mathcal{E} \cap \mathcal{F}] = \Pr[\mathcal{E}] \cdot \Pr[\mathcal{F}]$, from which the other equality holds as well.

It turns out to be a little cleaner to adopt this equivalent formulation as our working definition of independence. Formally, we'll say that events $\mathcal{E}$ and $\mathcal{F}$ are *independent* if $\Pr[\mathcal{E} \cap \mathcal{F}] = \Pr[\mathcal{E}] \cdot \Pr[\mathcal{F}]$.

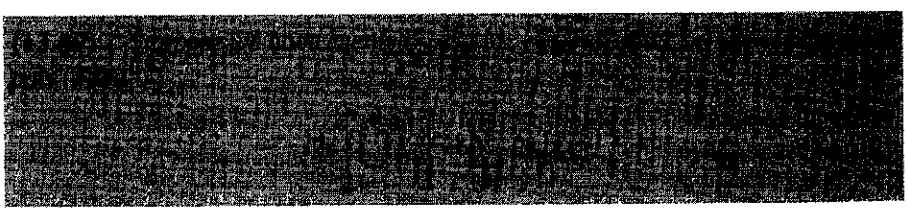
This product formulation leads to the following natural generalization. We say that a collection of events $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$ is *independent* if, for every set of indices $I \subseteq \{1, 2, \ldots, n\}$, we have

$$\Pr\left[\bigcap_{i \in I} \mathcal{E}_i\right] = \prod_{i \in I} \Pr[\mathcal{E}_i].$$

It's important to notice the following: To check if a large set of events is independent, it's not enough to check whether every pair of them is independent. For example, suppose we flip three independent fair coins: If $\mathcal{E}_i$ denotes the event that the $i^{\text{th}}$ coin comes up heads, then the events $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$ are independent and each has probability 1/2. Now let $A$ denote the event that coins 1 and 2 have the same value; let $B$ denote the event that coins 2 and 3 have the same value; and let $C$ denote the event that coins 1 and 3 have different values. It's easy to check that each of these events has probability 1/2, and the intersection of any two has probability 1/4. Thus every pair drawn from $A, B, C$ is independent. But the set of all three events $A, B, C$ is not independent, since $\Pr[A \cap B \cap C] = 0$.

## The Union Bound

Suppose we are given a set of events $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$, and we are interested in the probability that *any* of them happens; that is, we are interested in the probability $\Pr\left[\cup_{i=1}^{n} \mathcal{E}_i\right]$. If the events are all pairwise disjoint from one another, then the probability mass of their union is comprised simply of the separate contributions from each event. In other words, we have the following fact.



In general, a set of events $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$ may overlap in complex ways. In this case, the equality in (13.49) no longer holds; due to the overlaps among

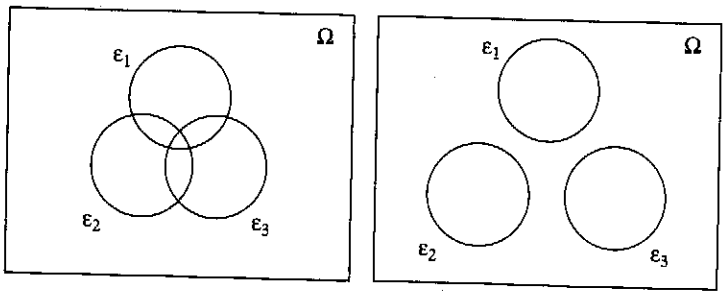



**Figure 13.5** The Union Bound: The probability of a union is maximized when the events have no overlap.

events, the probability mass of a point that is counted once on the left-hand side will be counted one *or more* times on the right-hand side. (See Figure 13.5.) This means that for a general set of events, the equality in (13.49) is relaxed to an inequality; and this is the content of the Union Bound. We have stated the Union Bound as (13.2), but we state it here again for comparison with (13.49).

Given its innocuous appearance, the Union Bound is a surprisingly powerful tool in the analysis of randomized algorithms. It draws its power mainly from the following ubiquitous style of analyzing randomized algorithms. Given a randomized algorithm designed to produce a correct result with high probability, we first tabulate a set of "bad events" $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$ with the following property: if none of these bad events occurs, then the algorithm will indeed produce the correct answer. In other words, if $\mathcal{F}$ denotes the event that the algorithm fails, then we have

$$\Pr[\mathcal{F}] \leq \Pr\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right].$$

But it's hard to compute the probability of this union, so we apply the Union Bound to conclude that

$$\Pr[\mathcal{F}] \leq \Pr\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right] \leq \sum_{i=1}^{n} \Pr[\mathcal{E}_i].$$

Now, if in fact we have an algorithm that succeeds with very high probability, and if we've chosen our bad events carefully, then each of the probabilities $\Pr[\mathcal{E}_i]$ will be so small that even their sum—and hence our overestimate of the failure probability—will be small. This is the key: decomposing a highly complicated event, the failure of the algorithm, into a horde of simple events whose probabilities can be easily computed.

Here is a simple example to make the strategy discussed above more concrete. Recall the Process Naming Problem we discussed earlier in this section, in which each of a set of processes chooses a random identifier. Suppose that we have 1,000 processes, each choosing a 32-bit identifier, and we are concerned that two of them will end up choosing the same identifier. Can we argue that it is unlikely this will happen? To begin with, let's denote this event by $\mathcal{F}$. While it would not be overwhelmingly difficult to compute $\Pr[\mathcal{F}]$ exactly, it is much simpler to bound it as follows. The event $\mathcal{F}$ is really a union of $\binom{1000}{2}$ "atomic" events; these are the events $\mathcal{E}_{ij}$ that processes $p_i$ and $p_j$ choose the same identifier. It is easy to verify that indeed, $\mathcal{F} = \cup_{i<j} \mathcal{E}_{ij}$. Now, for any $i \neq j$, we have $\Pr[\mathcal{E}_{ij}] = 2^{-32}$, by the argument in one of our earlier examples. Applying the Union Bound, we have

$$\Pr[\mathcal{F}] \leq \sum_{i,j} \Pr[\mathcal{E}_{ij}] = \binom{1000}{2} \cdot 2^{-32}.$$

Now, $\binom{1000}{2}$ is at most half a million, and $2^{32}$ is (a little bit) more than 4 billion, so this probability is at most $\frac{.5}{4000} = .000125$.

## Infinite Sample Spaces

So far we've gotten by with finite probability spaces only. Several of the sections in this chapter, however, consider situations in which a random process can run for arbitrarily long, and so cannot be well described by a sample space of finite size. As a result, we pause here to develop the notion of a probability space more generally. This will be somewhat technical, and in part we are providing it simply for the sake of completeness: Although some of our applications require infinite sample spaces, none of them really exercises the full power of the formalism we describe here.

Once we move to infinite sample spaces, more care is needed in defining a probability function. We cannot simply give each point in the sample space $\Omega$ a probability mass and then compute the probability of every set by summing. Indeed, for reasons that we will not go into here, it is easy to get into trouble if one even allows every subset of $\Omega$ to be an event whose probability can be computed. Thus a general probability space has three components:

(i) The sample space $\Omega$.

(ii) A collection $\mathcal{S}$ of subsets of $\Omega$; these are the only events on which we are allowed to compute probabilities.

(iii) A probability function Pr, which maps events in $\mathcal{S}$ to real numbers in $[0, 1]$.

The collection $\mathcal{S}$ of allowable events can be any family of sets that satisfies the following basic closure properties: the empty set and the full sample space $\Omega$ both belong to $\mathcal{S}$; if $\mathcal{E} \in \mathcal{S}$, then $\overline{\mathcal{E}} \in \mathcal{S}$ (closure under complement); and if $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, \ldots \in \mathcal{S}$, then $\cup_{i=1}^{\infty} \mathcal{E}_i \in \mathcal{S}$ (closure under countable union). The probability function Pr can be any function from $\mathcal{S}$ to $[0, 1]$ that satisfies the following basic consistency properties: $\Pr[\phi] = 0$, $\Pr[\Omega] = 1$, $\Pr[\mathcal{E}] = 1 - \Pr\left[\overline{\mathcal{E}}\right]$, and the Union Bound for disjoint events (13.49) should hold even for countable unions—if $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, \ldots \in \mathcal{S}$ are all pairwise disjoint, then

$$\Pr\left[\bigcup_{i=1}^{\infty} \mathcal{E}_i\right] = \sum_{i=1}^{\infty} \Pr\left[\mathcal{E}_i\right].$$

Notice how, since we are not building up Pr from the more basic notion of a probability mass anymore, (13.49) moves from being a theorem to simply a required property of Pr.

When an infinite sample space arises in our context, it's typically for the following reason: we have an algorithm that makes a sequence of random decisions, each one from a fixed finite set of possibilities; and since it may run for arbitrarily long, it may make an arbitrarily large number of decisions. Thus we consider sample spaces $\Omega$ constructed as follows. We start with a finite set of symbols $X = \{1, 2, \ldots, n\}$, and assign a *weight* $w(i)$ to each symbol $i \in X$. We then define $\Omega$ to be the set of all infinite sequences of symbols from $X$ (with repetitions allowed). So a typical element of $\Omega$ will look like $\langle x_1, x_2, x_3, \ldots \rangle$ with each entry $x_i \in X$.

The simplest type of event we will be concerned with is as follows: it is the event that a point $\omega \in \Omega$ begins with a particular finite sequence of symbols. Thus, for a finite sequence $\sigma = x_1 x_2 \ldots x_s$ of length $s$, we define the *prefix event associated with* $\sigma$ to be the set of all sample points of $\Omega$ whose first $s$ entries form the sequence $\sigma$. We denote this event by $\mathcal{E}_\sigma$, and we define its probability to be $\Pr[\mathcal{E}_\sigma] = w(x_1)w(x_2) \cdots w(x_s)$.

The following fact is in no sense easy to prove.

Once we have this fact, the closure of $\mathcal{S}$ under complement and countable union, and the consistency of Pr with respect to these operations, allow us to compute probabilities of essentially any "reasonable" subset of $\Omega$.

In our infinite sample space $\Omega$, with events and probabilities defined as above, we encounter a phenomenon that does not naturally arise with finite sample spaces. Suppose the set $X$ used to generate $\Omega$ is equal to $\{0, 1\}$, and $w(0) = w(1) = 1/2$. Let $\mathcal{E}$ denote the set consisting of all sequences that contain at least one entry equal to 1. (Note that $\mathcal{E}$ omits the "all-0" sequence.) We observe that $\mathcal{E}$ is an event in $\mathcal{S}$, since we can define $\sigma_i$ to be the sequence of $i-1$ 0s followed by a 1, and observe that $\mathcal{E} = \cup_{i=1}^{\infty} \mathcal{E}_{\sigma_i}$. Moreover, all the events $\mathcal{E}_{\sigma_i}$ are pairwise disjoint, and so

$$\Pr[\mathcal{E}] = \sum_{i=1}^{\infty} \Pr\left[\mathcal{E}_{\sigma_i}\right] = \sum_{i=1}^{\infty} 2^{-i} = 1.$$

Here, then, is the phenomenon: It's possible for an event to have probability 1 even when it's not equal to the whole sample space $\Omega$. Similarly, $\Pr\left[\overline{\mathcal{E}}\right] = 1 - \Pr[\mathcal{E}] = 0$, and so we see that it's possible for an event to have probability 0 even when it's not the empty set. There is nothing wrong with any of these results; in a sense, it's a necessary step if we want probabilities defined over infinite sets to make sense. It's simply that in such cases, we should be careful to distinguish between the notion that an event has probability 0 and the intuitive idea that the event "can't happen."

## Solved Exercises

### Solved Exercise 1

Suppose we have a collection of small, low-powered devices scattered around a building. The devices can exchange data over short distances by wireless communication, and we suppose for simplicity that each device has enough range to communicate with $d$ other devices. Thus we can model the wireless connections among these devices as an undirected graph $G = (V, E)$ in which each node is incident to exactly $d$ edges.

Now we'd like to give some of the nodes a stronger *uplink transmitter* that they can use to send data back to a base station. Giving such a transmitter to every node would ensure that they can all send data like this, but we can achieve this while handing out fewer transmitters. Suppose that we find a

subset $S$ of the nodes with the property that every node in $V - S$ is adjacent to a node in $S$. We call such a set $S$ a *dominating set*, since it "dominates" all other nodes in the graph. If we give uplink transmitters only to the nodes in a dominating set $S$, we can still extract data from all nodes: Any node $u \notin S$ can choose a neighbor $v \in S$, send its data to $v$, and have $v$ relay the data back to the base station.

The issue is now to find a dominating set $S$ of minimum possible size, since this will minimize the number of uplink transmitters we need. This is an NP-hard problem; in fact, proving this is the crux of Exercise 29 in Chapter 8. (It's also worth noting here the difference between dominating sets and vertex covers: in a dominating set, it is fine to have an edge $(u, v)$ with neither $u$ nor $v$ in the set $S$ as long as both $u$ and $v$ have neighbors in $S$. So, for example, a graph consisting of three nodes all connected by edges has a dominating set of size 1, but no vertex cover of size 1.)

Despite the NP-hardness, it's important in applications like this to find as small a dominating set as one can, even if it is not optimal. We will see here that a simple randomized strategy can be quite effective. Recall that in our graph $G$, each node is incident to exactly $d$ edges. So clearly any dominating set will need to have size at least $\frac{n}{d+1}$, since each node we place in a dominating set can take care only of itself and its $d$ neighbors. We want to show that a random selection of nodes will, in fact, get us quite close to this simple lower bound.

Specifically, show that for some constant $c$, a set of $\frac{cn \log n}{d+1}$ nodes chosen uniformly at random from $G$ will be a dominating set with high probability. (In other words, this completely random set is likely to form a dominating set that is only $O(\log n)$ times larger than our simple lower bound of $\frac{n}{d+1}$.)

***Solution*** Let $k = \frac{cn \log n}{d}$, where we will choose the constant $c$ later, once we have a better idea of what's going on. Let $\mathcal{E}$ be the event that a random choice of $k$ nodes is a dominating set for $G$. To make the analysis simpler, we will consider a model in which the nodes are selected one at a time, and the same node may be selected twice (if it happens to be picked twice by our sequence of random choices).

Now we want to show that if $c$ (and hence $k$) is large enough, then $\Pr[\mathcal{E}]$ is close to 1. But $\mathcal{E}$ is a very complicated-looking event, so we begin by breaking it down into much simpler events whose probabilities we can analyze more easily.

To start with, we say that a node $w$ *dominates* a node $v$ if $w$ is a neighbor of $v$, or $w = v$. We say that a set $S$ dominates a node $v$ if some element of $S$ dominates $v$. (These definitions let us say that a dominating set is simply a set of nodes that dominates every node in the graph.) Let $\mathcal{D}[v, t]$ denote the

event that the $t^{\text{th}}$ random node we choose dominates node $v$. The probability of this event can be determined quite easily: of the $n$ nodes in the graph, we must choose $v$ or one of its $d$ neighbors, and so

$$\Pr\left[\mathcal{D}[v,t]\right] = \frac{d+1}{n}.$$

Let $\mathcal{D}_v$ denote the event that the random set consisting of all $k$ selected nodes dominates $v$. Thus

$$\mathcal{D}_v = \bigcup_{t=1}^{k} \mathcal{D}[v,t].$$

For independent events, we've seen in the text that it's easier to work with intersections—where we can simply multiply out the probabilities—than with unions. So rather than thinking about $\mathcal{D}_v$, we'll consider the complementary "failure event" $\overline{\mathcal{D}_v}$, that no node in the random set dominates $v$. In order for no node to dominate $v$, each of our choices has to fail to do so, and hence we have

$$\overline{\mathcal{D}_v} = \bigcap_{t=1}^{k} \overline{\mathcal{D}[v,t]}.$$

Since the events $\overline{\mathcal{D}[v,t]}$ are independent, we can compute the probability on the right-hand side by multiplying all the individual probabilities; thus

$$\Pr\left[\overline{\mathcal{D}_v}\right] = \prod_{t=1}^{k} \Pr\left[\overline{\mathcal{D}[v,t]}\right] = \left(1 - \frac{d+1}{n}\right)^k.$$

Now, $k = \frac{cn \log n}{d+1}$, so we can write this last expression as

$$\left(1 - \frac{d+1}{n}\right)^k = \left[\left(1 - \frac{d+1}{n}\right)^{n/(d+1)}\right]^{c \log n} \leq \left(\frac{1}{e}\right)^{c \log n},$$

where the inequality follows from (13.1) that we stated earlier in the chapter.

We have not yet specified the base of the logarithm we use to define $k$, but it's starting to look like base $e$ is a good choice. Using this, we can further simplify the last expression to

$$\Pr\left[\overline{\mathcal{D}_v}\right] \leq \left(\frac{1}{e}\right)^{c \ln n} = \frac{1}{n^c}.$$

We are now very close to done. We have shown that for each node $v$, the probability that our random set fails to dominate it is at most $n^{-c}$, which we can drive down to a very small quantity by making $c$ moderately large. Now recall the original event $\mathcal{E}$, that our random set is a dominating set. This fails

to occur if and only if one of the events $\mathcal{D}_v$ fails to occur, so $\overline{\mathcal{E}} = \cup_v \overline{\mathcal{D}_v}$. Thus, by the Union Bound (13.2), we have

$$\Pr\left[\overline{\mathcal{E}}\right] \leq \sum_{v \in V} \Pr\left[\overline{\mathcal{D}_v}\right] \leq n \cdot \frac{1}{n^c} = \frac{1}{n^{c-1}}.$$

Simply choosing $c = 2$ makes this probability $\frac{1}{n}$, which is much less than 1. Thus, with high probability, the event $\mathcal{E}$ holds and our random choice of nodes is indeed a dominating set.

It's interesting to note that the probability of success, as a function of $k$, exhibits behavior very similar to what we saw in the contention-resolution example in Section 13.1. Setting $k = \Theta(n/d)$ is enough to guarantee that each individual node is dominated with constant probability. This, however, is not enough to get anything useful out of the Union Bound. Then, raising $k$ by another logarithmic factor is enough to drive up the probability of dominating each node to something very close to 1, at which point the Union Bound can come into play.

## Solved Exercise 2

Suppose we are given a set of $n$ variables $x_1, x_2, \ldots, x_n$, each of which can take one of the values in the set $\{0, 1\}$. We are also given a set of $k$ equations; the $r^{\text{th}}$ equation has the form

$$(x_i + x_j) \bmod 2 = b_r$$

for some choice of two distinct variables $x_i, x_j$, and for some value $b_r$ that is either 0 or 1. Thus each equation specifies whether the sum of two variables is even or odd.

Consider the problem of finding an assignment of values to variables that maximizes the number of equations that are satisfied (i.e., in which equality actually holds). This problem is NP-hard, though you don't have to prove this.

For example, suppose we are given the equations

$$(x_1 + x_2) \bmod 2 = 0$$

$$(x_1 + x_3) \bmod 2 = 0$$

$$(x_2 + x_4) \bmod 2 = 1$$

$$(x_3 + x_4) \bmod 2 = 0$$

over the four variables $x_1, \ldots, x_4$. Then it's possible to show that no assignment of values to variables will satisfy all equations simultaneously, but setting all variables equal to 0 satisfies three of the four equations.

(a) Let $c^*$ denote the maximum possible number of equations that can be satisfied by an assignment of values to variables. Give a polynomial-time algorithm that produces an assignment satisfying at least $\frac{1}{2}c^*$ equations. If you want, your algorithm can be randomized; in this case, the *expected* number of equations it satisfies should be at least $\frac{1}{2}c^*$. In either case, you should prove that your algorithm has the desired performance guarantee.

(b) Suppose we drop the condition that each equation must have exactly two variables; in other words, now each equation simply specifies that the sum of an arbitrary subset of the variables, mod 2, is equal to a particular value $b_r$.

Again let $c^*$ denote the maximum possible number of equations that can be satisfied by an assignment of values to variables, and give a polynomial-time algorithm that produces an assignment satisfying at least $\frac{1}{2}c^*$ equations. (As before, your algorithm can be randomized.) If you believe that your algorithm from part (a) achieves this guarantee here as well, you can state this and justify it with a proof of the performance guarantee for this more general case.

***Solution*** Let's recall the punch line of the simple randomized algorithm for MAX 3-SAT that we saw earlier in the chapter: If you're given a constraint satisfaction problem, assigning variables at random can be a surprisingly effective way to satisfy a constant fraction of all constraints.

We now try applying this principle to the problem here, beginning with part (a). Consider the algorithm that sets each variable independently and uniformly at random. How well does this random assignment do, in expectation? As usual, we will approach this question using linearity of expectation: If $X$ is a random variable denoting the number of satisfied equations, we'll break $X$ up into a sum of simpler random variables.

For some $r$ between 1 and $k$, let the $r^{\text{th}}$ equation be

$$(x_i + x_j) \bmod 2 = b_r.$$

Let $X_r$ be a random variable equal to 1 if this equation is satisfied, and 0 otherwise. $E[X_r]$ is the probability that equation $r$ is satisfied. Of the four possible assignments to equation $i$, there are two that cause it to evaluate to 0 mod 2 ($x_i = x_j = 0$ and $x_i = x_j = 1$) and two that cause it to evaluate to 1 mod 2 ($x_i = 0; x_j = 1$ and $x_i = i; x_j = 0$). Thus $E[X_r] = 2/4 = 1/2$.

Now, by linearity of expectation, we have $E[X] = \sum_r E[X_r] = k/2$. Since the maximum number of satisfiable equations $c^*$ must be at most $k$, we satisfy at least $c^*/2$ in expectation. Thus, as in the case of MAX 3-SAT, a simple random assignment to the variables satisfies a constant fraction of all constraints.

For part (b), let's press our luck by trying the same algorithm. Again let $X_r$ be a random variable equal to 1 if the $r^{\text{th}}$ equation is satisfied, and 0 otherwise; let $X$ be the total number of satisfied equations; and let $c^*$ be the optimum.

We want to claim that $E[X_r] = 1/2$ as before, even when there can be an arbitrary number of variables in the $r^{\text{th}}$ equation; in other words, the probability that the equation takes the correct value mod 2 is exactly 1/2. We can't just write down all the cases the way we did for two variables per equation, so we will use an alternate argument.

In fact, there are two natural ways to prove that $E[X_r] = 1/2$. The first uses a trick that appeared in the proof of (13.25) in Section 13.6 on hashing: We consider assigning values arbitrarily to all variables but the last one in the equation, and then we randomly assign a value to the last variable $x$. Now, regardless of how we assign values to all other variables, there are two ways to assign a value to $x$, and it is easy to check that one of these ways will satisfy the equation and the other will not. Thus, regardless of the assignments to all variables other than $x$, the probability of setting $x$ so as to satisfy the equation is exactly 1/2. Thus the probability the equation is satisfied by a random assignment is 1/2.

(As in the proof of (13.25), we can write this argument in terms of conditional probabilities. If $\mathcal{E}$ is the event that the equation is satisfied, and $\mathcal{F}_b$ is the event that the variables other than $x$ receive a sequence of values $b$, then we have argued that $\Pr[\mathcal{E} \mid \mathcal{F}_b] = 1/2$ for all $b$, and so $\Pr[\mathcal{E}] = \sum_b \Pr[\mathcal{E} \mid \mathcal{F}_b] \cdot \Pr[\mathcal{F}_b] = (1/2) \sum_b \Pr[\mathcal{F}_b] = 1/2$.)

An alternate proof simply counts the number of ways for the $r^{\text{th}}$ equation to have an even sum, and the number of ways for it to have an odd sum. If we can show that these two numbers are equal, then the probability that a random assignment satisfies the $r^{\text{th}}$ equation is the probability it gives it a sum with the right even/odd parity, which is 1/2.

In fact, at a high level, this proof is essentially the same as the previous one, with the difference that we make the underlying counting problem explicit. Suppose that the $r^{\text{th}}$ equation has $t$ terms; then there are $2^t$ possible assignments to the variables in this equation. We want to claim that $2^{t-1}$ assignments produce an even sum, and $2^{t-1}$ produce an odd sum, which will show that $E[X_r] = 1/2$. We prove this by induction on $t$. For $t = 1$, there are just two assignments, one of each parity; and for $t = 2$, we already proved this earlier by considering all $2^2 = 4$ possible assignments. Now suppose the claim holds for an arbitrary value of $t - 1$. Then there are exactly $2^{t-1}$ ways to get an even sum with $t$ variables, as follows:

- $2^{t-2}$ ways to get an even sum on the first $t - 1$ variables (by induction), followed by an assignment of 0 to the $t^{\text{th}}$, plus

- $2^{t-2}$ ways to get an odd sum on the first $t-1$ variables (by induction), followed by an assignment of 1 to the $t^{\text{th}}$.

The remaining $2^{t-1}$ assignments give an odd sum, and this completes the induction step.

Once we have $E[X_r] = 1/2$, we conclude as in part (a): Linearity of expectation gives us $E[X] = \sum_r E[X_r] = k/2 \geq c^*/2$.

## Exercises

1. *3-Coloring* is a yes/no question, but we can phrase it as an optimization problem as follows.

   Suppose we are given a graph $G = (V, E)$, and we want to color each node with one of three colors, even if we aren't necessarily able to give different colors to every pair of adjacent nodes. Rather, we say that an edge $(u, v)$ is *satisfied* if the colors assigned to $u$ and $v$ are different.

   Consider a 3-coloring that maximizes the number of satisfied edges, and let $c^*$ denote this number. Give a polynomial-time algorithm that produces a 3-coloring that satisfies at least $\frac{2}{3}c^*$ edges. If you want, your algorithm can be randomized; in this case, the *expected* number of edges it satisfies should be at least $\frac{2}{3}c^*$.

2. Consider a county in which 100,000 people vote in an election. There are only two candidates on the ballot: a Democratic candidate (denoted $D$) and a Republican candidate (denoted $R$). As it happens, this county is heavily Democratic, so 80,000 people go to the polls with the intention of voting for $D$, and 20,000 go to the polls with the intention of voting for $R$.

   However, the layout of the ballot is a little confusing, so each voter, independently and with probability $\frac{1}{100}$, votes for the wrong candidate—that is, the one that he or she *didn't* intend to vote for. (Remember that in this election, there are only two candidates on the ballot.)

   Let $X$ denote the random variable equal to the number of votes received by the Democratic candidate $D$, when the voting is conducted with this process of error. Determine the expected value of $X$, and give an explanation of your derivation of this value.

3. In Section 13.1, we saw a simple distributed protocol to solve a particular contention-resolution problem. Here is another setting in which randomization can help with contention resolution, through the distributed construction of an independent set.